

An exact algorithm with the time complexity of $O^*(1.299^m)$ for the weighed mutually exclusive set cover problem

Songjian Lu and Xinghua Lu

Department of Biomedical Informatics,
University of Pittsburgh, Pittsburgh, PA 15219, USA
Email: songjian@pitt.edu, xinghua@pitt.edu

Abstract

In this paper, we will introduce an exact algorithm with a time complexity of $O^*(1.299^m)^\dagger$ for the WEIGHTED MUTUALLY EXCLUSIVE SET COVER problem, where m is the number of subsets in the problem. This problem has important applications in recognizing mutation genes that cause different cancer diseases.

1 Introduction

The SET COVER problem is that: given a ground set X of n elements and a collection \mathcal{F} of m subsets of X , try to find a minimum number of subsets S_1, S_2, \dots, S_h in \mathcal{F} such that $\cup_{i=1}^h S_i = X$. If we add an additional constrain such that all subsets in the solution are pairwise disjoint, then the SET COVER problem becomes the MUTUALLY EXCLUSIVE SET COVER problem. If we further assign each subset in \mathcal{F} a real number weight and search the solution with the minimum weight, i.e. the sum of weights of subsets in the solution is minimized, then the problem becomes the WEIGHTED MUTUALLY EXCLUSIVE SET COVER problem.

Recently, the WEIGHTED MUTUALLY EXCLUSIVE SET COVER problem has found important applications in cancer study to identify driver mutations [4, 12], i.e. somatic mutations that cause cancers. As somatic mutations will change the structures (and therefore the functions) of signaling proteins; thus, perturb cancer pathways that regulate the expressions of genes in certain important biological processes, such as cell death, cell proliferation etc. The perturbations within a common cancer pathway are often found to be mutually exclusive in a single cancer cell, i.e. each tumor usually has only one perturbation on one given cancer pathways (one perturbation is enough to cause the disease; hence, there is no need to wait for another perturbation). Modern lab techniques can identify somatic mutations and gene expressions of cancer cells. After preprocessing the data, we will obtain following information for important biological processes, e.g. cell death: 1) which cancer cells have disturbed the expressions of genes in the biological process; 2) which genes have been mutated in those cancer cells; 3) how possible each mutation is related to the given biological process (i.e. each mutation is assigned a real number weight). Then next step is finding a set of mutations such that each cancer cell has one and only one mutation in the solution set (mutually exclusive) and the sum of weights of all genes in the solution set is minimized, which is the WEIGHTED MUTUALLY EXCLUSIVE SET COVER problem.

While there is not much research on the MUTUALLY EXCLUSIVE SET COVER or the WEIGHTED MUTUALLY EXCLUSIVE SET COVER problems, the SET COVER problem has been paid much attention. The SET COVER, which is equivalent to the HITTING SET problem, is a fundamental NP-hard problem in Karp's

[†]**Note:** Following the recent convention, we use a star $*$ to represent that the polynomial part of the time complexity is neglected.

21 NP-complete problems [8]. One research direction for the SET COVER problem is approximation algorithms, e.g. papers [1, 5, 9, 11] gave polynomial time approximation algorithms that find solutions whose sizes are at most $c \log n$ times the size of the optimal solution, where c is a constant. Second direction is using k , the number of subsets in the solution, as parameter to design fixed-parameter tractable (FPT) algorithms for the equivalent problem, the HITTING SET problem. Those algorithms have a constrain such that each element in X is included in at most d subsets in \mathcal{F} , i.e. sizes of all subsets in the HITTING SET problem are upper bound by d ; it is also called the d -HITTING SET problem. For example, paper [13] gave an $O^*(2.270^k)$ algorithm for the 3-HITTING SET problem, and paper [6] further improved the time complexity to $O^*(2.179^k)$. The third direction is designing algorithms that use n as parameter in the condition that n is much less than m . Papers [2, 7] designed algorithms with time complexities of $O^*(2^n)$ for the problem. The paper [2] also extended the algorithm to solve the WEIGHTED MUTUALLY EXCLUSIVE SET COVER problem with the same time complexity. Paper [10] improved the time complexity to $O^*(2^{\frac{\log_2 d}{1+\log_2 d} n})$ under the condition that at least $\frac{n}{1+\log_2 d}$ elements in X are included in at most d subsets in \mathcal{F} . This algorithm can also be extended to the WEIGHTED MUTUALLY EXCLUSIVE SET COVER problem with the same time complexity. However, in the application of cancer study, neither n is less than m nor each element in x is included in bounded number of subsets in \mathcal{F} . Hence, there is a need to design new algorithms.

In this paper, we will design a new algorithm that uses m as parameter (in application of cancer study, m is smaller than n , where n can be as large as several hundreds). Trivially, if using m as parameter, we can solve the problem in time of $O^*(2^m)$, where the algorithm basically just tests every combination of subsets in \mathcal{F} . To our best knowledge, we have not found any algorithm that is better than the trivial algorithms when using m as parameter. This paper will give the first un-trivial algorithm with the time complexity of $O^*(1.299^m)$ to solve the WEIGHTED MUTUALLY EXCLUSIVE SET COVER problem. We have tested this algorithm in the cancer study, and the program can finish the computation practically when m is less than 100.

2 The WEIGHTED MUTUALLY EXCLUSIVE SET COVER problem is NP-hard

The formal definition of the WEIGHTED MUTUALLY EXCLUSIVE SET COVER problem is: given a ground set X of n elements, a collection \mathcal{F} of m subsets of X , and a weight function $w : \mathcal{F} \rightarrow [0, \infty)$, if $\mathcal{F}' = \{S_1, S_2, \dots, S_h\} \subset \mathcal{F}$ such that $\cup_{i=1}^h S_i = X$, and $S_i \cap S_j = \emptyset$ for any $i \neq j$, then we say \mathcal{F}' is a mutually exclusive set cover of X and $\sum_{i=1}^h w(S_i)$ is the weight of \mathcal{F}' ; the goal of the problem is to find a mutually exclusive set cover of X with the minimum weight, or report that no such solution exists.

As we have not found the proof of NP-hardness for the WEIGHTED MUTUALLY EXCLUSIVE SET COVER problem, in this section, we will prove that the MUTUALLY EXCLUSIVE SET COVER problem is NP-hard; thus, prove that the WEIGHTED MUTUALLY EXCLUSIVE SET COVER problem is NP-hard.

We will prove the NP-hardness of the MUTUALLY EXCLUSIVE SET COVER problem by reducing another NP-hard problem, the MAXIMUM SET PACKING problem, to it. Remember that the MAXIMUM SET PACKING problem is: given a collection \mathcal{F} of subsets, try to find an $\mathcal{S} \subset \mathcal{F}$ such that subsets in \mathcal{S} are pairwise disjoint and $|\mathcal{S}|$ is maximized.

Theorem 2.1 *The MUTUALLY EXCLUSIVE SET COVER problem is NP-hard.*

PROOF. Let $\mathcal{S} = \{S_1, S_2, \dots, S_m\}$ be an instance of the MAXIMUM SET PACKING problem, where $X' = \cup_{i=1}^m S_i = \{x_1, x_2, \dots, x_n\}$. We create an instance of the MUTUALLY EXCLUSIVE SET COVER problem such that:

- $X = X' \cup \{T_1, T_2, \dots, T_m\}$, where $T_i = \{t_{i1}, t_{i2}, \dots, t_{i(n+1)}\}$ for all $1 \leq i \leq m$;
- $\mathcal{F} = \mathcal{F}' \cup \mathcal{F}'' \cup \mathcal{F}'''$, where $\mathcal{F}' = \{\{x_1\}, \{x_2\}, \dots, \{x_n\}\}$, $\mathcal{F}'' = \{S_1 \cup T_1, S_2 \cup T_2, \dots, S_m \cup T_m\}$, and $\mathcal{F}''' = \cup_{i=1}^m \{\{t_{i1}\}, \{t_{i2}\}, \dots, \{t_{i(n+1)}\}\}$.

Next, we will prove that if $\mathcal{P} = \{P_1, P_2, \dots, P_k\}$ is a solution of the MUTUALLY EXCLUSIVE SET COVER problem, then $\mathcal{S}' = \{S'_1, S'_2, \dots, S'_{k'}\}$ is a solution of the MAXIMUM SET PACKING problem, where $\mathcal{P} \cap \mathcal{F}'' = \{S'_1 \cup T'_1, S'_2 \cup T'_2, \dots, S'_{k'} \cup T'_{k'}\}$. Thus we will prove that the time to solve the MAXIMUM SET PACKING problem is bounded by the total time of transforming the MAXIMUM SET PACKING problem into the MUTUALLY EXCLUSIVE SET COVER, and of solving the MUTUALLY EXCLUSIVE SET COVER problem. Therefore, the MUTUALLY EXCLUSIVE SET COVER problem is NP-hard.

As subsets in \mathcal{P} are pairwise disjoint, it is obvious that subsets in \mathcal{S}' are pairwise disjoint. Hence, if we suppose that \mathcal{S}' is not the solution of the MAXIMUM SET PACKING problem, then there must exists a $\mathcal{S}'' = \{S''_1, S''_2, \dots, S''_{k'}\} \subset \mathcal{S}$ such that subsets in \mathcal{S}'' are pairwise disjoint and $k' > k$. Thus we can make a new solution \mathcal{P}' of the MUTUALLY EXCLUSIVE SET COVER problem such that \mathcal{P}' includes $\{S''_1 \cup T''_1, S''_2 \cup T''_2, \dots, S''_{k'} \cup T''_{k'}\} \subset \mathcal{F}''$ and other subsets in \mathcal{F}' and \mathcal{F}''' . If let $|X' - \cup_{i=1}^k S'_i| = n_1$ and $|X' - \cup_{i=1}^{k'} S''_i| = n_2$ (Note: any T_i , which is not covered by a subset in \mathcal{F}'' , needs $n + 1$ subsets in \mathcal{F}''' to cover it; any $x_i \in X'$, which is not covered by a subset in \mathcal{F}'' , needs a subset in \mathcal{F}' to cover it), then

$$|\mathcal{P}| = k + (m - k)(n + 1) + n_1,$$

and

$$|\mathcal{P}'| = k' + (m - k')(n + 1) + n_2.$$

Therefore $|\mathcal{P}| - |\mathcal{P}'| = (k' - k)n + n_1 - n_2 > 0$, i.e. \mathcal{P}' is a solution with less subsets in \mathcal{F} , which cases contradiction that \mathcal{P} is the solution of the MUTUALLY EXCLUSIVE SET COVER problem. Hence, \mathcal{S}' is a solution of the MAXIMUM SET PACKING problem. \square

3 The main Algorithm

In this section, we will introduce our new algorithm to solve the WEIGHTED MUTUALLY EXCLUSIVE SET COVER problem.

Let (X, \mathcal{F}, w) be an instance of the WEIGHTED MUTUALLY EXCLUSIVE SET COVER problem. We can use a bipartite graph to represent (X, \mathcal{F}, w) such that all nodes on one sides are subsets in \mathcal{F} while nodes on the other side are elements in X , and if an element u of X is in subset U , i.e. $u \in U$, then an edge is added between u and U . For the convenience, let us introduce some notations. The Figure 1 can help you to understand and remember following notations.

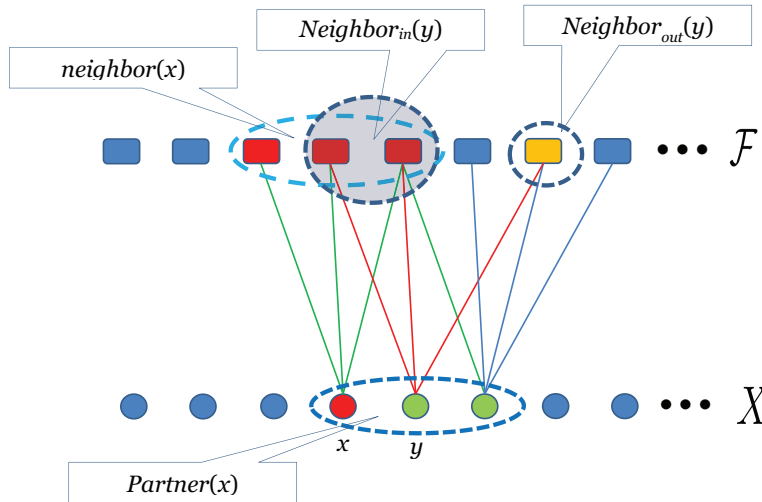


Figure 1: Graph representation and some notations of the problem

For any $x \in X$, let $neighbor(x) = \{S | S \in \mathcal{F} \text{ and } x \in S\}$, $degree(x) = |neighbor(x)|$, $partner(x) = \cup_{S \in neighbor(x)} S$. For any y in $partner(x)$, let $neighbor_{in} = neighbor(y) \cap neighbor(x)$, $degree_{in}(y) = |neighbor_{in}(y)|$, $neighbor_{out} = neighbor(y) - neighbor(x)$, $degree_{out}(y) = |neighbor_{out}(y)|$.

Algorithm-1 WMES-Cover((X, \mathcal{F}, w), $Solution_{partial}$, $Solution_{final}$)

Input: An instance of the WEIGHTED MUTUALLY EXCLUSIVE SET COVER problem, two variables,
where $Solution_{final}$ is a global variable to keep the best solution.

Output: A minimum weight mutually exclusive set cover or “No Solution”.

```

1  if  $X == \emptyset$  then
1.1  if  $weight(Solution_{partial}) < weight(Solution_{final})$  then replace  $Solution_{final}$  with  $Solution_{partial}$ ;
2  Find  $x \in X$  such that  $d = degree(x)$  is minimized;
3  if  $d == 0$  then return “No Solution”;
4  if  $d == 1$  then WMES-Cover( $(X - \{x\}, \mathcal{F} - neighbor(x), w)$ ,  $Solution_{partial} \cup neighbor(x)$ ,  $Solution_{final}$ );
5  if  $degree_{out}(y) == 0$  for all  $y \in partner(x)$  then
5.1  if there exists  $S \in neighbor(x)$  such that  $S == partner(x)$  then
5.1.1  WMES-Cover( $(X - S, \mathcal{F} - neighbor(x), w)$ ,  $Solution_{partial} \cup \{S\}$ ,  $Solution_{final}$ );
      else
5.1.2  return “No Solution”;
6  if  $d == 2$  then // Suppose  $neighbor(x) = \{S_1, S_2\}$ ; note that  $S_1 \subset X$  and  $S_2 \subset X$ .
6.1  WMES-Cover( $(X - S_1, \mathcal{F} - \cup_{u \in S_1} neighbor(u), w)$ ,  $Solution_{partial} \cup \{S_1\}$ ,  $Solution_{final}$ );
6.2  WMES-Cover( $(X - S_2, \mathcal{F} - \cup_{u \in S_2} neighbor(u), w)$ ,  $Solution_{partial} \cup \{S_2\}$ ,  $Solution_{final}$ );
      else // (Note:  $d > 2$ )
6.3  if there exists a  $y \in partner(x)$  such that  $degree_{out}(y) = 1$  then
6.3.1  Let  $y \in partner(x)$  such that  $degree_{out}(y) = 1$  and  $W' \in neighbor_{out}(y)$ ;
6.3.2  if  $|neighbor(x) - neighbor(y)| > 0$  then // (Note:  $|neighbor(x) - neighbor(y)| \leq 1$ )
6.3.2.1  Find any  $W \in neighbor(x) - neighbor(y)$ ;
6.3.2.2  WMES-Cover( $(X - W' \cup W, \mathcal{F} - \cup_{u \in W' \cup W} neighbor(u), w)$ ,  $Solution_{partial} \cup \{W', W\}$ ,  $Solution_{final}$ );
6.3.2.3  WMES-Cover( $(X, \mathcal{F} - \{W', W\}, w)$ ,  $Solution_{partial}$ ,  $Solution_{final}$ );
      else
6.3.2.4  Find any  $W \in neighbor(x)$ ;
6.3.2.5  WMES-Cover( $(X - W, \mathcal{F} - \cup_{u \in W} neighbor(u), w)$ ,  $Solution_{partial} \cup \{W\}$ ,  $Solution_{final}$ );
6.3.2.6  WMES-Cover( $(X, \mathcal{F} - \{W', W\}, w)$ ,  $Solution_{partial}$ ,  $Solution_{final}$ );
      else
6.3.3  Find a  $y \in partner(x)$  such that  $degree_{out}(y)$  is maximized;
6.3.4  Find a  $Z \in neighbor_{in}(y)$ ;
6.3.5  WMES-Cover( $(X - Z, \mathcal{F} - \cup_{u \in Z} neighbor(u), w)$ ,  $Solution_{partial} \cup \{Z\}$ ,  $Solution_{final}$ );
6.3.6  WMES-Cover( $(X, \mathcal{F} - \{Z\}, w)$ ,  $Solution_{partial}$ ,  $Solution_{final}$ );

```

Figure 2: Algorithm for the WEIGHTED MUTUALLY EXCLUSIVE SET COVER problem.

The main algorithm, Algorithm-1, is shown in Figure 2. Basically, the Algorithm-1 first finds an $x \in X$ with minimum degree and then branches at one subset in $neighbor(x)$ (such as in step 6.2.2 and 6.2.3). For the convenience, if $degree(x) = d$, then we say that Algorithm-1 is doing a d -branch. Because of steps 3,4,5, when the program arrives at step 6, we must have: 1) $d = degree(x) \geq 2$; 2) for any $u \in X$, $degree(u) \geq d$; 3) there exists a $y \in partner(x)$ such that $degree_{out}(y) > 0$.

The Algorithm-1 is basically searching the solution by going through a search tree; hence, if knowing the number of leaves in the search tree, then we will obtain the time complexity of the Algorithm-1. Next, we will estimate the number of leaves in the search tree by studying the different cases of branching. We begin from the 2-branch.

Proposition 3.1 *The search tree has at most 1.273^m leaves If only the 2-branches are applied in Algorithm-1.*

PROOF. Suppose that $degree(x) = 2$ and $y \in partner(x)$ such that $degree_{out}(y) > 0$. Let $neighbor(x) = \{S_1, S_2\}$.

In the case of $\text{degree}_{\text{out}}(y) = 1$, let $\text{neighbor}_{\text{out}}(y) = \{S''\}$. In the branches of choosing either S_1 or S_2 into the solution, if y is covered, then S'' will be removed from the \mathcal{F} , or else if y is not covered yet, then S'' will be chosen into the solution in order to cover y (note: after S_1, S_2 are removed, $\text{degree}(y) = 1$ in the new instance (at line 6.1.1 and 6.1.2 of Algorithm-1); thus, S'' will be included into the solution in the next call of the Algorithm-1 in this branch). Hence, in any case, 3 subsets in \mathcal{F} will be removed. If letting $T(k)$ be the number of leaves in the search tree when $|\mathcal{F}| = k$, then we will obtain the following recurrence relation

$$T(k) \leq 2T(k-3). \quad (1)$$

The characteristic equation of this recurrence relation is $r^3 - 2 = 0$ [‡]; hence, we will have $T(m) < 1.260^m$.

In the case of $\text{degree}_{\text{out}}(y) > 1$, we consider following sub-cases.

Sub-case 1. Suppose $\text{degree}_{\text{in}}(y) = 1$, and $y \in S_1$. Then at least S_1 and S_2 will be removed from \mathcal{F} for the branch of choosing S_2 into the solution; at least S_1, S_2 , and all subsets (at least two) in $\text{neighbor}_{\text{out}}(y)$ will be removed for the branch of choosing S_1 into the solution. Thus the recurrence relation of $T(k)$ is

$$T(k) \leq T(k-2) + T(k-4). \quad (2)$$

which leads to $T(m) < 1.273^m$.

Sub-case 2. Suppose $\text{degree}_{\text{in}}(y) = 2$. Then in either branch, y is covered by S_1 or S_2 , which is chosen into the solution. Hence, S_1, S_2 , and all subsets (at least two) in $\text{neighbor}_{\text{out}}(y)$ will be removed from \mathcal{F} . Thus we will obtain the recurrence relation

$$T(k) \leq 2T(k-4). \quad (3)$$

which leads to $T(m) < 1.190^m$.

By considering all above cases, we obtain that $T(m) \leq 1.273^m$. □

Now, we consider the case of doing 3-branch. Remember that when Algorithm-1 is doing a 3-branch, $\text{degree}(x) \geq 3$ for all $x \in X$.

Proposition 3.2 *The search tree has at most 1.299^m leaves If only the d -branches for $d \leq 3$ are applied in Algorithm-1.*

PROOF. The cases of 2-branches are considered in the last proposition. Now we consider the cases of 3-branches. Suppose that $\text{degree}(x) = 3$ and $y \in \text{partner}(x)$ such that $\text{degree}_{\text{out}}(y) > 0$. Let $\text{neighbor}(x) = \{S_1, S_2, S_3\}$.

If $\text{degree}_{\text{out}}(y) = 1$, then $\text{degree}_{\text{in}}(y) \geq 2$ (as $\text{degree}(y) \geq 3$). Let $\{S'\} = \text{neighbor}_{\text{out}}(y)$. We further consider following sub-cases.

Sub-case 1. Suppose $\text{degree}_{\text{in}}(y) = 2$. Let $S_1 \in \text{neighbor}(x) - \text{neighbor}(y)$. The Algorithm-1 branches at S_1 . The branch one includes S_1 into the solution; thus, S_2, S_3 will be removed. This will further make $\text{degree}(y) = 1$. Hence, S' will also be included into the solution. Totally, in this branch, we will remove at least 4 subsets from \mathcal{F} . In branch two, we will exclude S_1 from the solution. Then either S_2 or S_3 must be included into the solution. Thus y is covered by S_2 or S_3 , and S' will not be in the solution. Therefore, in this branch, we know that at least S_1 and S' will be removed. So we will obtain the recurrence relation

$$T(k) \leq T(k-2) + T(k-4), \quad (4)$$

which leads to $T(m) < 1.273^m$.

[‡]**Note:** Given a recurrence relation $T(k) \leq \sum_{i=0}^{k-1} c_i T(i)$ such that all c_i are nonnegative real numbers, $\sum_{i=0}^{k-1} c_i > 0$, and $T(0)$ represents the leaves, then $T(k) \leq r^k$, where r is the unique positive root of the characteristic equation $t^k - \sum_{i=0}^{k-1} c_i t^i = 0$ deduced from the recurrence relation [3].

Sub-case 2. Suppose $\text{degree}_{in}(y) = 3$. Then S' will not in the solution and any one of S_1, S_2, S_3 (one and only one of them must be included into the solution to cover x) will cover y . The Algorithm-1 will branch at any one of S_1, S_2, S_3 . Without loss of generality, we branch at S_1 . In the branch of including S_1 into the solution, S_1, S_2, S_3 will be removed, which will totally remove at least 4 subsets. In the branch of excluding S_1 into the solution, S_1 will be removed. Thus 2 subsets will be removed. We will obtain the following recurrence relation

$$T(k) \leq T(k-2) + T(k-4), \quad (5)$$

which leads to $T(m) < 1.273^m$.

In the case of $\text{degree}_{out}(y) > 1$, Let $S_1 \in \text{neighbor}_{in}(y)$. Algorithm-1 branches at S_1 . In the first branch, S_1 is included into the solution. Then S_1, S_2, S_3 and at least 2 subsets in $\text{neighbor}_{out}(y)$ will be removed. In the second branch, S_1 is excluded, which will make $\text{degree}(x) = 2$ in the new instance; hence, in this branch, a 2-branch will follow. Thus even considering the worst case of the 2-branch (the recurrence relation (2)), we will have

$$T(k) \leq 2T(k-5) + T(k-3), \quad (6)$$

which will lead to $T(m) \leq 1.299^m$.

From all above cases and Proposition 3.1, we will have $T(m) \leq 1.299^m$. \square

Let us consider the case of doing d -branch for $d > 3$.

Proposition 3.3 *The search tree in Algorithm-1 has at most 1.299^m leaves.*

PROOF. We only need to consider the cases of d -branches for $d > 3$. Suppose that $\text{degree}(x) = d$ and $y \in \text{partner}(x)$ such that $\text{degree}_{out}(y) > 0$. Let $\text{neighbor}(x) = \{S_1, S_2, \dots, S_d\}$.

In the case of $\text{degree}_{out}(y) = 1$, $\text{degree}_{in}(y)$ can only be $d-1$ or d .

Sub-case 1. Suppose $\text{degree}_{in}(y) = d-1$. Then there is one and only one subset in $\text{neighbor}(x) - \text{neighbor}_{in}(y)$. Without loss of generality, we suppose $S_1 \notin \text{neighbor}_{in}(y)$. Algorithm-1 will branch on S_1 such that in the branch of including S_1 into the solution, all d subsets in $\text{neighbor}(x)$ and one subset in $\text{neighbor}_{out}(y)$ will be removed (i.e. in this branch, at least 5 subsets will be removed; in the branch of excluding S_1 from the solution, one subset in $\{S_2, S_3, \dots, S_d\}$ will be included into the solution, which y will be covered and the only subset in $\text{neighbor}_{out}(y)$ will be removed (i.e. in this branch, two subsets will be removed). Therefore, we will have following recurrence relation

$$T(k) \leq T(k-5) + T(k-2), \quad (7)$$

which leads to $T(m) < 1.237^m$.

Sub-case 2. Suppose $\text{degree}_{in}(y) = d$. Without loss of generality, we suppose that Algorithm-1 branches on S_1 . Then it is easy to understand the we will have the following recurrence relation

$$T(k) \leq T(k-5) + T(k-2), \quad (8)$$

which leads to $T(m) < 1.237^m$.

In the case of $\text{degree}_{out}(y) > 1$, suppose $S_1 \in \text{degree}_{in}(y)$ and Algorithm-1 branches on S_1 . Then in the branch of including S_1 into the solution, all subsets in $\text{neighbor}(x)$ and $\text{neighbor}_{out}(y)$ will be removed (at least 6 subsets will be removed). In the branch of excluding S_1 into the solution, at least one subset S_1 will be removed. Hence, we will have the recurrence relation

$$T(k) \leq T(k-6) + T(k-1), \quad (9)$$

which leads to $T(m) < 1.286^m$.

Considering all above cases, Proposition 3.1, and Proposition 3.2, we have $T(m) \leq 1.299^m$. \square

Theorem 3.4 *The WEIGHTED MUTUALLY EXCLUSIVE SET COVER problem can be solved by an algorithm with a time complexity of $O^*(1.299^m)$.*

PROOF. Let (\mathcal{F}, X, w) be an instance of the WEIGHTED MUTUALLY EXCLUSIVE SET COVER problem, where X is a ground set of n elements, \mathcal{F} is a collection of m subsets of X , and $w : \mathcal{F} \rightarrow [0, \infty)$ is the weight function. Now we prove that the problem can be solved by the Algorithm-1 in time $O^*(1.299^m)$.

The correctness of the algorithm is easy to understand. If there is an $x \in X$ such that $\text{degree}(x) = 0$, then x cannot be covered by any subset in \mathcal{F} . Thus, the problem has no solution. The step 3 of the Algorithm-1 deals with this situation. If, for any given $x \in X$, $\text{degree}(x) = 1$, then there exists one and only one subset in \mathcal{F} that covers x , i.e. $\text{neighbor}(x)$ must be included into the solution. Thus x and $\text{neighbor}(x)$ will be removed from the problem. This situation is dealt with in step 4. If for all y in $\text{partner}(x)$, $\text{degree}_{\text{out}}(y) = 0$, then $\text{partner}(x)$ can only be covered by subset(s) in $\text{neighbor}(x)$. By the exclusivity, at most one subset in $\text{neighbor}(x)$ can be chosen into the solution. Thus, if finding a subset S in $\text{neighbor}(x)$ such that $S = \text{partner}(x)$, then Algorithm-1 will include S into the solution, or else the problem has no solution. The step 5 of the Algorithm-1 deals with this situation.

After the Algorithm-1 reaches step 6, we have: 1) for all $x' \in X$, $\text{degree}(x') \geq \text{degree}(x) > 1$ (as x is the element in X with the minimum degree); 2) there is a $y \in \text{partner}(x)$ such that $\text{degree}_{\text{out}}(y) > 0$. If $d = \text{neighbor}(x) = 2$, then one and only one subset in $\text{neighbor}(x)$ will be in the solution. The step 6.1 and 6.2 correctly deals with this situation. For the cases after step 6.2, the Algorithm-1 basically chooses one subset S in $\text{neighbor}(x)$ and branches on S such that one branch includes S into the solution and the other branch excludes S from the solution (Note: when $\text{degree}_{\text{out}}(y) = 1$, we used a small trick to include or exclude the additional subset in $\text{neighbor}_{\text{out}}(y)$ into or from the solution; please refer to sub-case 1 and sub-case 2 in the Proposition 3.3). Therefore, Algorithm-1 will go through the search tree and find the solution with the minimum weight (if the solution exists), which is saved in step 1.1.

By Proposition 3.3, the search tree has at most 1.299^m leaves. Hence, the time complexity of the algorithm is bounded by $O^*(1.299^m)$. If we further notice that the time to process each node is bounded by $O(mn)$, then the more accurate time complexity of the algorithm is $O(1.299^m mn)$. \square

4 Problem extension

In this paper, we first proved that the WEIGHTED MUTUALLY EXCLUSIVE SET COVER problem is NP-hard. Then we designed the first non-trivial algorithm, which uses the m as parameter, with a time complexity of $O^*(1.352^m)$ for the problem. the WEIGHTED MUTUALLY EXCLUSIVE SET COVER problem has been used to find the driver mutations in cancers [4, 12]. Our new algorithm can find the optimal solution for the problem, which is better than solutions found by the heuristic algorithms in the previous research [4, 12]. The exclusivity is the extreme case. In practical applications, a cancer cell may have more than one mutation to perturb a common pathway. Hence, a modified model is finding a set of mutations with minimum weight sum such that each cancer cell has at least one and at most t ($t=2$ or 3) mutations in the solutions, which leads to the SMALL OVERLAPPED SET COVER problem. Also, on application, some mutations in cancer cells may not be detected because of errors. Thus, it is not always ideal to find a solution mutations that cover all cancer cells. A modified model is finding a set of mutually exclusive mutations that cover at least r percent (90% or 95%) of cancer cells, which leads to the MAXIMAL SET COVER problem. Our next research will design efficient algorithms for above two new problems.

References

- [1] N. Alon, D. Moshkovitz, and S. Safra, Algorithmic Construction of Sets for k -Restrictions, ACM Transaction on Algorithms, 2(2), pp. 153-177, 2006.

- [2] A. Björlund, T. Husfeldt, M. Koivisto, Set partitioning via Inclusion-Exclusion. *SIAM Journal on Computing*, Special Issue for FOCS 2006.
- [3] J. Chen, I. Kanj, and W. Jia, Vertex Cover: Further Observations and Further Improvements, *Journal of Algorithm*, 41, pp. 280-301, 2001.
- [4] G. Ciriello, E. Cerami, C. Sander, N. Schultz, Mutual exclusivity analysis identifies oncogenic network modules, *Genome research*, 22(2), pp. 398-406, 2012.
- [5] U. Feige, A Threshold of $\ln n$ for Approximation Set Cover, *J. of the ACM*, 45(4), pp. 634-652, 1998.
- [6] H. Fernau, a top-down approach to search-trees: Improved algorithmics for 3-Hitting Set, *Algorithmica*, 57, pp. 97-118, 2010.
- [7] Q. Hua, Y. Wang, D. Yu, F. Lau, Dynamic programming based algorithms for set multicover and multiset multicover problem. *Theoretical Computer Science V411*, pp. 2467-2474, 2010.
- [8] R. Karp, Reducibility Among Combinatorial Problems, In R. E. Miller and J. W. Thatcher (editors). *Complexity of Computer Computations*. New York: Plenum, pp. 85-103, 1972.
- [9] S. Kolliopoulos, N. Young, Approximation algorithms for covering/packing integer programs. *J. Comput. Syst. Sci.* 71(4), pp.495-505, 2005.
- [10] S. Lu, X. Lu, A graph model and an exact algorithm for finding transcription factor modules, 2nd ACM Conference on Bioinformatics, Computational Biology and Biomedicine, pp. 355-359, 2011.
- [11] C. Lund, and M. Yannakakis, On the Hardness of Approximating Minimization Problem, *J. of the Association for Computing Machinery*, 45(5), pp. 960-981, 1994.
- [12] C. Miller, S. Settle, E. Sulman, K. Aldape, A. Milosavljevic, Discovering functional modules by identifying recurrent and mutually exclusive mutational patterns in tumors, *BMC medical genomics*, 4, pp. 34, 2011.
- [13] R. Niedermeier, and P. Rossmanith, An Efficient Fixed-parameter Algorithm for 3-Hitting Set, *J. of Discrete Algorithms*, 1(1), pp. 89-102, 2003.